Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Matic Kladnik

Večnivojska spletna vizualizacija zbirk zvočnih posnetkov

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: doc. dr. Matija Marolt

Ljubljana, 2016

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V okviru diplomske naloge izdelajte sistem za večnivojsko vizualizacijo zbirke zvočnih posnetkov. Na vsakem pogledu prikažite pregledno količino zvokov, za kar uporabite večnivojsko gručenje na podlagi podobnosti zvokov. Podobnosti izračunajte z uporabo značilk, ki zajemajo različne aspekte barve zvoka, uporabnik pa naj ima nadzor nad tem kateri aspekti so pomembni za prikaz zbirke.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Matic Kladnik sem avtor diplomskega dela z naslovom:

Večnivojska spletna vizualizacija zbirk zvočnih posnetkov (angl. *A multi-level web-based visualization of sound collections*)

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Matije Marolta,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 16. marca 2016 Podpis avtorja:

Zahvaljujem se mentorju doc. dr. Matiji Maroltu in Pii Jerkič. Zahvaljujem se moji družini za vso podporo pri mojem študiju.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
dB	decibel	decibel
HFC	High Frequency Content	visokofrekvenčna vsebina
HSL	Hue Saturation Lightness	odtenek, intenzivnost, svetlost
HTML	HyperText Markup Language	jezik za označevanje nadbesedila
JSON	JavaScript Object Notation	zapis objekta v Javascript
MDS	Multi-Dimensional Scaling	večrazsežnostno lestvičenje
MFCC	Mel Frequency Cepstral Coefficients	-
RGB	Red Green Blue	rdeča, zelena, modra
SVG	Scalable Vector Graphics	stopnjevane vektorske slike

Povzetek

Naslov: Večnivojska spletna vizualizacija zbirk zvočnih posnetkov

Zvočne zbirke so v vsakdanjem življenju vedno pogostejše. Poraja se vprašanje, kako uporabniku prikazati veliko zvočno zbirko, da mu bodo v pregledni obliki dostopni vsi potrebni podatki. Tema diplomske naloge je izdelava sistema za spletno vizualizacijo večjih zbirk zvočnih podatkov. Za vizualizacijo smo uporabili dvonivojsko metodologijo, kar pomeni da lahko prehajamo med gručami zvočnih podatkov na dveh nivojih. Gručenje zvočnih posnetkov poteka glede na podobnosti zvokov med seboj; za izračun podobnosti smo uporabili značilke, pridobljene s knjižnico Essentia. Prednost uporabe naše dvonivojske metodologije je, da lahko pregledno analiziramo podatke, ter podrobno preiskujemo podatkovno zbirko.

Ključne besede: vizualizacija, zbirka, podatki, zvok, analiza, gručenje.

Abstract

Title: A multi-level web-based visualization of sound collections

Sound collections are becoming more frequently used in our daily lives. A common dilemma is how to present a large sound collection in a concise way but at the same time enable user access to all relevant data. Main objective of this thesis is creation of the system for concise visualization of collections of sound data. To achieve that we have used a two-level methodology for visualization, which means we can move between data clusters on two levels. Sound records are clustered based on similarity between them. For computation of similarities we have used attributes that are retrieved from Essentia library. The advantage of using our methodology is the ability to view data in a manageable way and research the data collection in greater detail due to the multi-level design.

Keywords: visualization, collection, data, sound, analysis, clustering.

Poglavje 1

Uvod

Namen dela je ustvariti sistem za spletno vizualizacijo zbirk zvočnih podatkov, ki bo za uporabnike intuitiven, ter s katerim se bo lahko upravljalo z majhnim številom potez. V sami diplomski nalogi bo predstavljen rezultat, torej sama vizualizacija, ter algoritmi in orodja s katerimi je vizualizacija bila ustvarjena.

Vizualizacija podatkov je bila uporabljena že skozi različna obdobja v zgodovini, pa naj bo to v grafični ali slikovni obliki. Že stoletja se ljudje za razumevanje zanašajo na vizualizacijo podatkov, bodisi s histogrami, načrti, tabelami, ali kakšnim drugim načinom vizualne predstavitve podatkov. Vizualizacija uporabnikom omogoča lažji in predvsem hitrejši pregled podatkov na podlagi določenih parametrov. Zato je smiselna izdelava vizualizacije, ki bi bila uporabnikom dostopna na spletu.

Na svetu je prisotnih vedno več zvočnih podatkov, ki se zbirajo v zbirke. Zbirke se uporabljajo v storitvah za: prepoznavo skladb, v storitvah za pretočno predvajanje glasbe, za industrijske, filmske potrebe in še marsikje. Z vizualizacijo teh podatkov je možno precej izboljšati naše razumevanje podatkov [3]. Porajajo se tudi vprašanja, kako te velike količine podatkov ustrezno združiti in predstaviti preučevalcu.

Cilj diplomske naloge je implementacija sistema za vizualizacijo večje zbirke zvočnih podatkov, tako da bo ta razločno razčlenjena in enostavna

za uporabo. V ta namen smo razvili dvonivojsko vizualizacijo, ki deluje po principu združevanja zvokov v skupine in podskupine, s čimer dosežemo dvonivojsko hierarhičnost. Pri tem je prehod med nivojema možen le z enim klikom na elemente vizualizacij. Skupine zvokov so ustrezno ločene, tako da je pregled zbirke čim enostavnejši. Z vizualizacijo je možna interakcija oziroma upravljanje zgolj s kliki na posamezne komponente. Zaradi enostavnosti upravljanja je vizualizacijo možno pregledati tudi z mobilnimi napravami.

Za implementacijo našega sistema smo uporabili zbirko zvočnih posnetkov projekta Work With Sounds. Ker so vizualizacije namenjene lažjemu preučevanju podatkov, je bilo poleg same implementacije potrebno vložiti precej časa tudi v testiranje prototipov, ter končnega izdelka. Sam cilj smo dosegli z lastno implementacijo sistema za vizualizacijo zvočnih podatkov, pri katerem smo uporabili programsko knjižnico za izris vektorske grafike na spletu, ter več programskih knjižnic za zaledni sistem, namenjen obdelavi zvočnih podatkov.

V naslednjem poglavju bomo pregledali: področje vizualizacij zbirk podatkov, tako zvočnih podatkov, kot tudi na splošno, pogosto uporabljene načine za združevanje podatkov, ter nekaj praktičnih primerov vizualizacij.

Poglavje 2

Pregled področja vizualizacij podatkov

Zvočne zbirke lahko vsebujejo toliko podatkov, da jih naenkrat ne moremo razločno predstaviti na zaslonu. Zato je smiselno te podatke razvrstiti v skupine podatkov, ki so si po lastnostih podobni. Med temi lastnostmi sta lahko ritem, kompleksnost, ali kaj podobnega. Večje zbirke je smiselno razčleniti na podskupine, saj lahko tako zbirko bolj razločno prikažemo. Naenkrat prikazujemo le podatke ene ali nekaj skupin, kar omogoči pregledno vizualizacijo podatkov, ki jo lahko uporabnik pregleduje na več nivojih.

Vizualizacija lahko ugodno vpliva tudi na hitrost človeške obravnave podatkov in zniža nivo zahtevanega znanja uporabnika. S pomočjo ogleda neke grafike, ki predstavlja bolj zapleteno informacijo, se človeška kognitivna obremenjenost nekoliko zmanjša v primerjavi z direktno analizo podatkov [31]. Interaktivna vizualizacija je cenjeno orodje tudi za analizo podatkov [7].

Vizualizacija je v zvočnih zbirkah prisotna na različne načine, denimo z virtualnimi površinami, na katerih lahko uporabnik v tridimenzionalnem svetu pregleda zvočne zapise [19]. Kot močan koncept pridobivanja informacij se je izkazala tudi vizualizacija rezultatov v projektu Three Georges, kjer so vizualizirali podatke o ekološko-okoljskih vplivih [40]. Vizualizacija je lahko ključna tudi pri predstavitvi rezultatov podatkovnega rudarjenja

[18]. Prav tako pri vizualni analizi velikih grafov, ki so rezultat metod umetne inteligence [2]. Vizualizacije so ključne tudi za hitro analizo dinamičnega združevanja v skupine [15]. To področje je odprto še za številne spremembe, pri čemer vizualizacija zelo pospeši načrtovanje sprememb, denimo pri združevanju velikih skupin podatkov [4]. Vizualizacija je uporabna tudi pri prikazu in primerjavi rezultatov pridobivanja znanja iz teksta ali druge vrste podatkov [6]

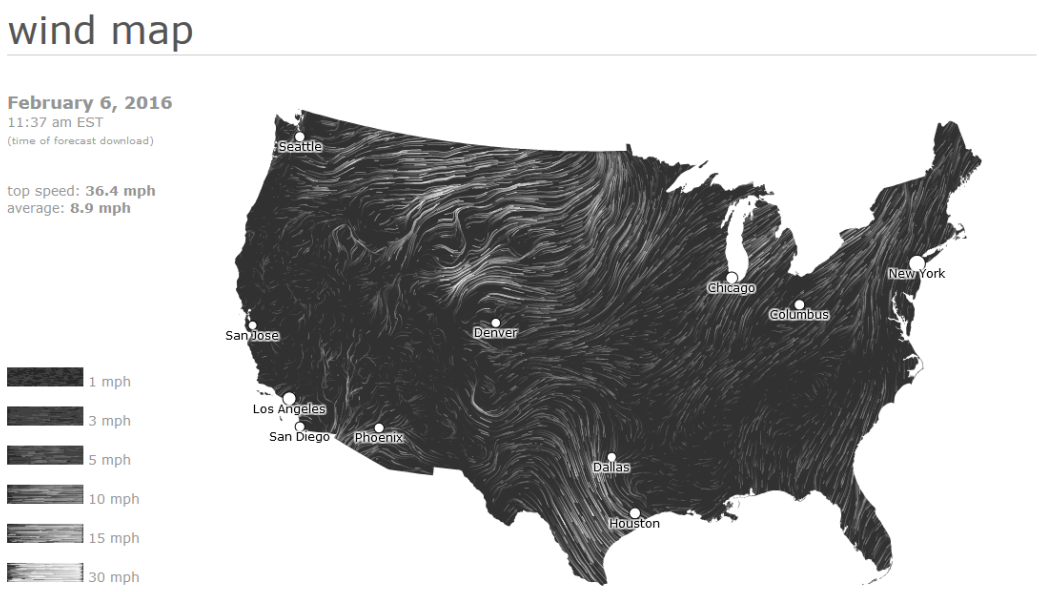
Razvrščanja v skupine in dinamične mreže so prav tako fleksibilna, kar pomeni da se spreminjajo skozi čas [39]. Zato je potrebno vsake spremembe in nove podatke ustrezno umestiti. Študija pojasnjuje [39], da imamo ljudje podobno razumevanje glasbenega tempa in višine glasbe, tako da lahko umestimo vsak posnetek na 2D površino glede na vrednosti tempa in višine. Še posebej pa je področje vizualizacije podatkov dobro izkoriščeno na spletu, saj so tako podatki dostopni večji množici ljudi [8].

2.1 Pregled in komentar nekaj primerov vizualizacij

V tem podpoglavju si bomo pogledali nekaj zanimivih primerov vizualizacij, ki jih lahko najdemo na svetovnem spletu.

2.1.1 Vetrovna mapa ZDA

Prvi primer [16] prikazuje smer in hitrost vetra v Združenih državah Amerike. Ta primer sem vključil, ker prikazuje smiselnost uporabe vizualizacij kot sredstev za analizo informacij in podatkov, s pomočjo katerih lahko lažje sprejemamo odločitve. Na tej vizualizaciji je moč celovito spremljati, kako se gibljejo vetrovi po celotnem področju ZDA. Vizualizacija omogoča smiseln pregled tokov, ter uporabnikovo interakcijo za podrobnejše spremljanje manjšega dela geografskega področja. Zaradi zveznega prikaza tokov je moč zelo dobro opazovati gibanje zračnih tokov na širšem področju.

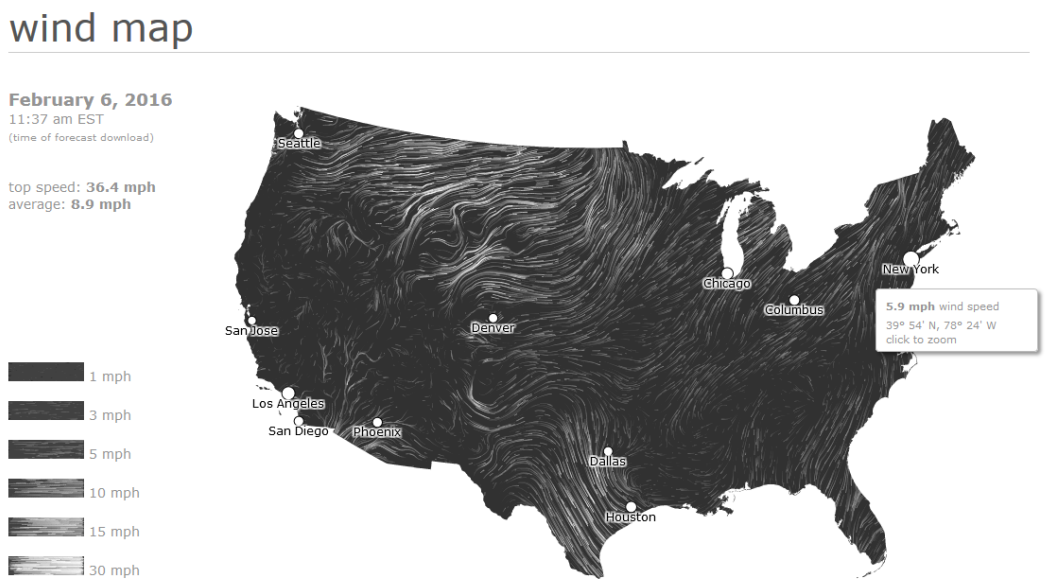


Slika 2.1: začetna zaslonska slika vizualizacije vetrov v ZDA

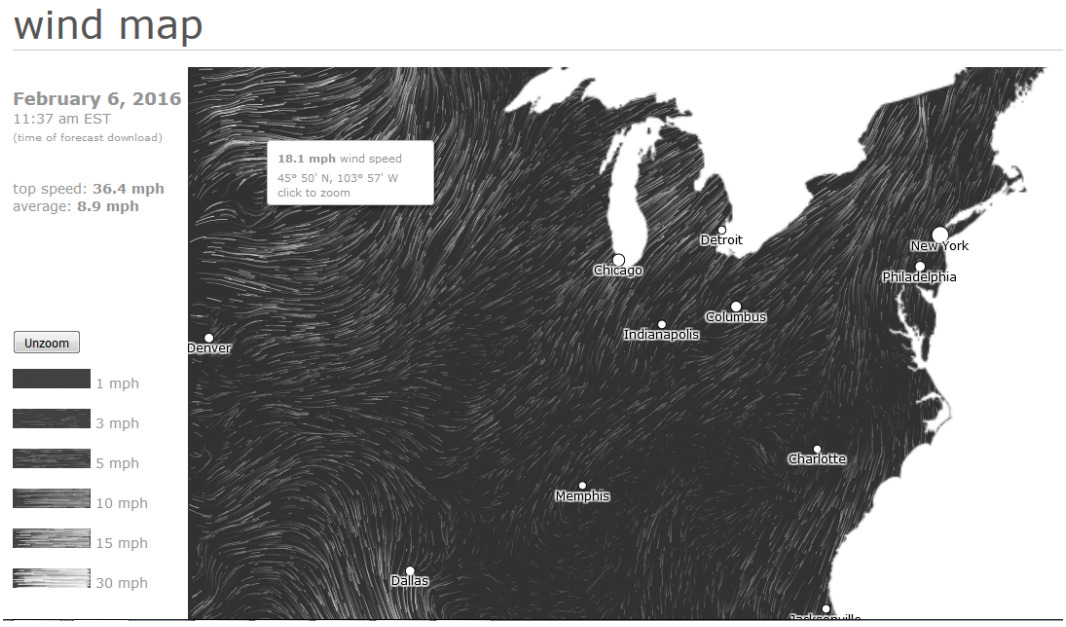
Manjša pomanjkljivost vizualizacije je nekoliko nekonsistentno upravljanje s povečavo vizualizacije. Za povečavo je potreben le klik na mapi, za pomanjšavo slike pa klik na poseben gumb, viden na sliki 2.3 nad legendo hitrosti vetrov.

2.1.2 Spoznavanje strojnega učenja

Da lahko s pomočjo vizualizacije lažje in celoviteje širimo znanje, lahko vidimo na naslednjem primeru vizualnega uvoda v strojno učenje, dostopnega na [32]. Ta vizualizacija je vključena v pregled področja za prikaz uporabnosti vizualizacij za kvalitetnejše širjenje znanja. Na spletni strani lahko uporabniki s pomočjo nekaj vizualizacij pregledajo, kako iz danih podatkov določiti pripadnost stanovanja enemu od mest. Stanovanja lahko pripadajo New Yorku ali San Franciscu. Skozi primere se uporabnik uči o klasifikacijah, klasifikacijskih drevesih, učnem modelu in nekaterih drugih metodah strojnega učenja. Ker so vizualizacije dinamične, je bolj preprosto razumeti



Slika 2.2: začetna slika z dodatnimi podatki o izbranem toku vetra



Slika 2.3: povečava na ožje področje v okolici New Yorka

Slika 2.4: zaslonska slika predstavitve problema s pomočjo vizualizacije

kako poteka klasifikacija stanovanj v klasifikacijskem drevesu, poleg tega pa vsaka vizualizacija predstavi svoj vidik učne snovi. Na ta način se tekst in vizualizacije lepo dopolnjujejo. Predstavniki stanovanj iz San Francisca so obarvani zeleno, predstavniki stanovanj iz New Yorka pa modro. Ta barvna razdelitev je uporabljena na vseh vizualizacijah tega učnega vodiča.

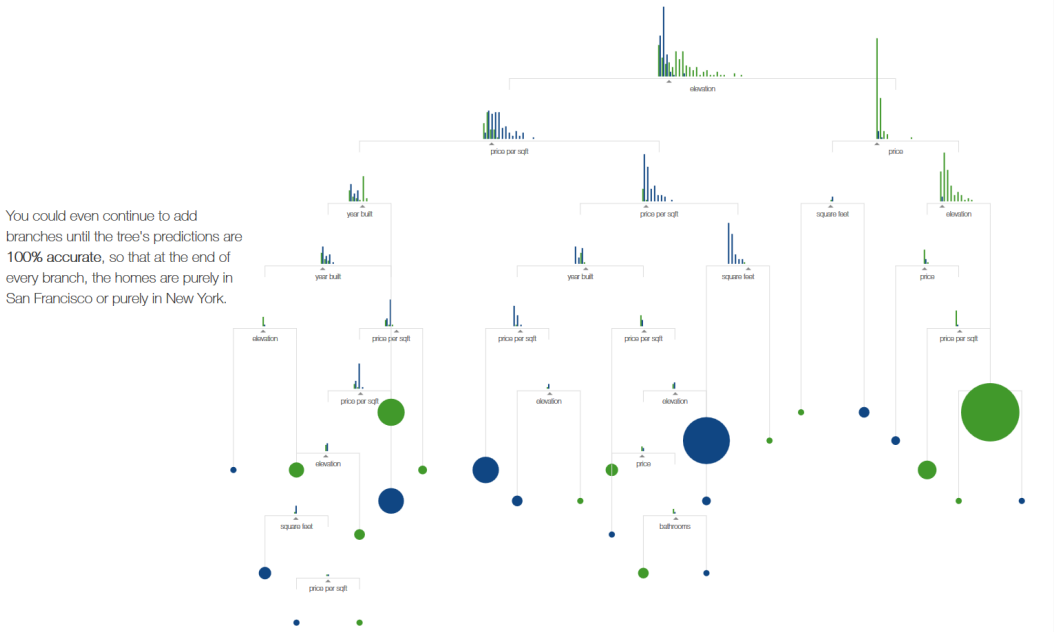
2.2 Pregled primerov vizualizacij zvočnih zbirk

V tem podpoglavju bomo pregledali nekaj primerov vizualizacij zvočnih zbirk.

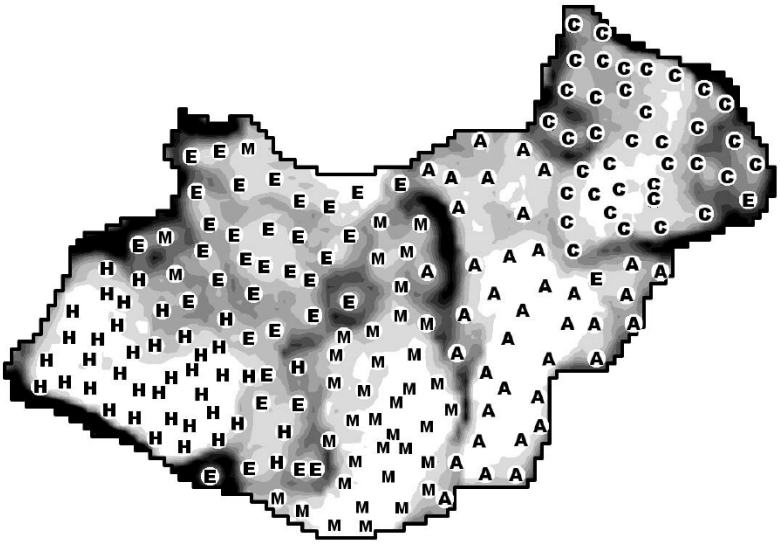
2.2.1 Vizualizacija zvočne zbirke glede na razdalje v percepciji

Skupina raziskovalcev univerze Philipps-University v Marburgu je izvedla vizualizacijo za sistem za organiziranje večjih zvočnih zbirk. Projekt je po-

Slika 2.5: zaslonska slika predstavitve klasifikacije glede na višino in ceno stanovanja



Slika 2.6: zaslonska slika predstavitve klasifikacijskega drevesa s pomočjo vizualizacije

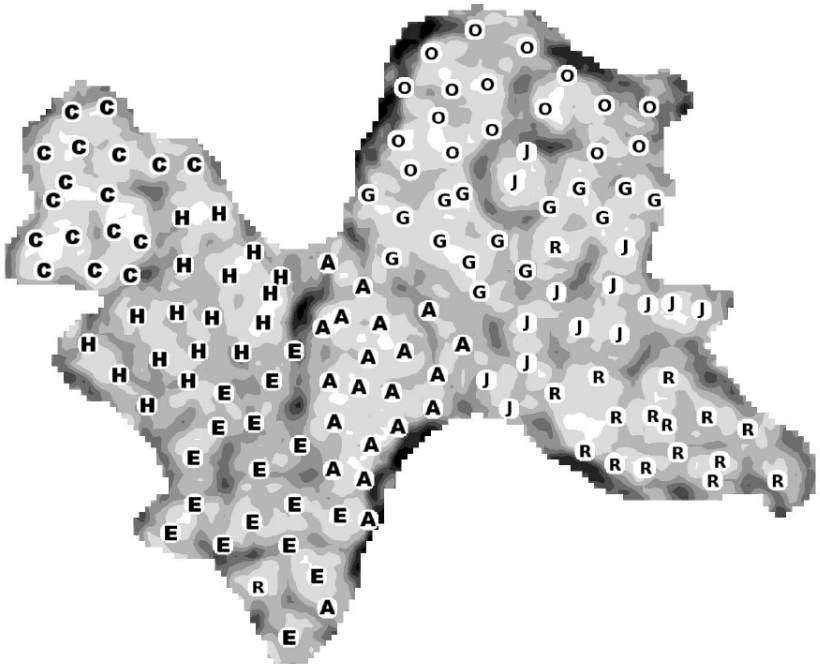


Slika 2.7: vizualizacija zvokov v pet skupin; M=metal/rok, A=akustika, C=klasična glasba, H=hiphop, E=elektronska glasba

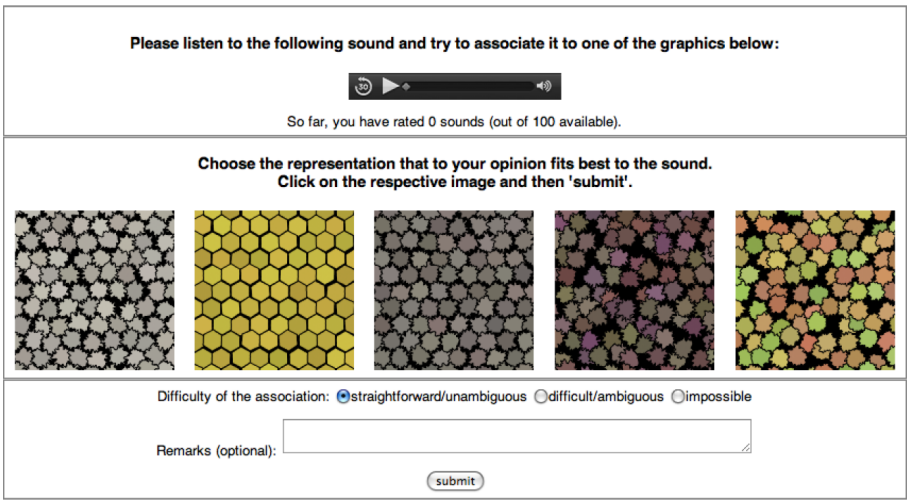
drobneje opisan v članku [21]. Sistem MusicMiner deluje na podlagi tehnik podatkovno-bioničnega rudarjenja. V gručah se pojavijo podobno zveneči zvoki. Za vizualizacijo na sliki 2.7 so razporedili 200 skladb v 8 različnih glasbenih žanrov. Ti so: akustična glasba, klasična glasba, hiphop, metal/rok in elektronska glasba.

Za vizualizacijo na sliki 2.8 so raziskovalci razporedili 140 zvokov v 8 različnih glasbenih žanrov. Ti so: alternativen rok, stand-up komedija, nemški hiphop, elektronska glasba, jazz, oldies, opera in reggae.

Temnejša področja na vizualizaciji predstavljajo daljše razdalje v podatkovnem prostoru. Vizualizaciji dobro ločita zvoke in prikažeta sestavo zbirke. Vendar ne vemo katere zvoke oziroma skladbe predstavljajo točke na vizualizaciji, kar je glavna pomanjkljivost te vizualizacije. Vizualizacija je enonivojska, tako da se lahko na tej vizualizaciji pregledno prikaže precej manj zvočnih podatkov, kot na dvonivojski.



Slika 2.8: vizualizacija zvokov v osem skupin; A=alternativen rok, C=komedija, H=hiphop, E=elektronska glasba, J=jazz, G=oldies, O=opera in R=reggae

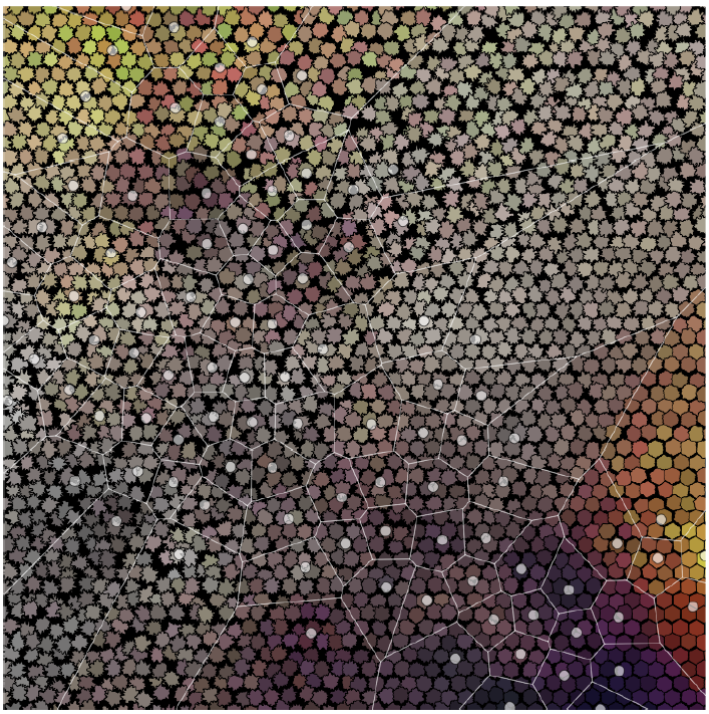


Slika 2.9: spletni vmesnik s predvajalnikom za asociacijo zvoka z eno izmed prikazanih tekstur

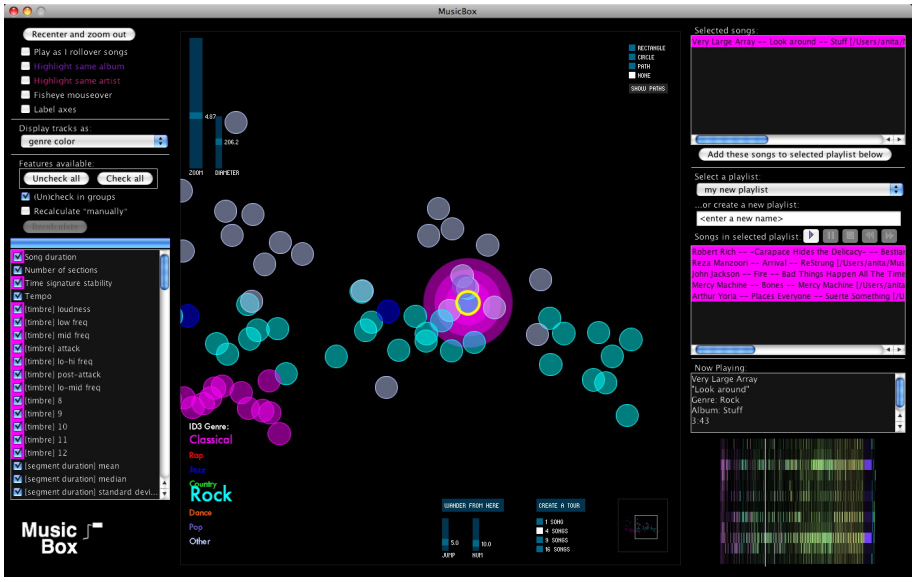
2.2.2 Vizualizacija percepcije zvokov s teksturami

V tej vizualizaciji uporabnik na spletnem vmesniku za vsakega izmed zvokov v zbirki izbere teksturo, ki se mu zdi najprimernejša za predvajani zvok. Prototip je delo raziskovalcev avstrijskega inštituta za umetno inteligenco OFAI z Dunaja. Projekt je opisan v članku [14]. Osnova za velik del vizualnih predstavitev zvokov so nizkonivojski fizikalni parametri, kot so spektralni koeficienti. Slabost takih vizualizacij je, da so precej abstraktne, saj težko opišejo človeško percepcijo zvoka [14]. V ta namen se s to vizualizacijo na podlagi uporabnikovih izbir direktno povezuje zvoke s teksturami.

Uporabnik lahko pregledno povezuje zvoke v zbirki s teksturami, ki se mu zdijo najprimernejše. Možen je pregled celotne zbirke zvokov s pomočjo izbranih tekstur, kot je vidno na sliki 2.10. Pomanjkljivost vizualizacije je, da se zvoke gruči le na podlagi uporabnikove vizualne interpretacije zvoka, torej se gručenje opravi le na podlagi uporabnikove percepcije zvoka. Ta vizualizacija je enonivojska, tako da ni možno prikazati tako velike zbirke zvočnih podatkov, kot na večnivojski vizualizaciji.



Slika 2.10: spletni pregled zbirke zvokov s teksturami

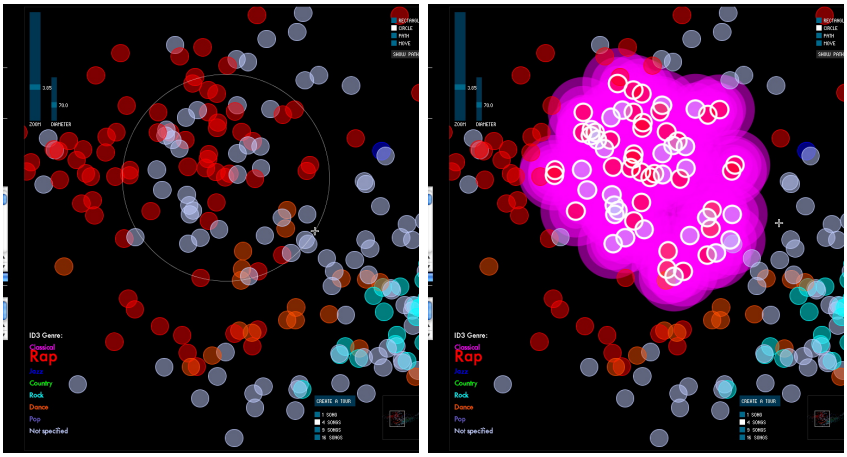


Slika 2.11: zaslonska slika vizualizacije MusicBox

2.2.3 Vizualizacija zvočnih zbirk MusicBox

V tej vizualizaciji lahko pregledujemo zbirko zvočnih posnetkov, razporejenih po površini glede na vrednosti zvočnih značilk. Tudi ta vizualizacija je enonivojska. Točke, ki predstavljajo zvoke so obarvane glede na pripadnosti glasbenim žanrom (slika 2.11). To omogoča preverjanje podobnosti predstavnikov različnih žanrov. Kot je vidno na slikah 2.12, lahko izberemo več zvokov naenkrat, ti pa se dodajo na listo za predvajanje.

V vizualizaciji MusicBox je možno tudi približevanje in oddaljevanje pogleda, tako da je lahko naenkrat na oknu vizualizacije vidna celotna zbirka ali pa le del nje. S tem orodjem je možno nekoliko bolj pregledno preiskovati zvočne zbirke, vendar še vedno manj pregledno kot na večnivojskih vizualizacijah. Še vedno se namreč dogaja, da so točke podobnih zvokov lahko razporejene na majhni površini, ter se zaradi tega prekrivajo. Zato je analiza večje zbirke zvočnih podatkov težja, kot na dvonivojskih vizualizacijah. Podrobnejši opis vizualizacije je dostopen na spletni strani avtorice [20].



(a) izbira zvokov

(b) prikaz obarvanosti izbranih zvokov

Slika 2.12: zaslonski sliki vizualizacije med izbiro zvokov (vir: [20])

V tem poglavju smo pregledali nekaj zanimivih vrst vizualizacij, ki se uporabljajo na različnih področjih vsakdanjega življenja in so primerne za prikaz na spletnih brskalnikih.

Poglavje 3

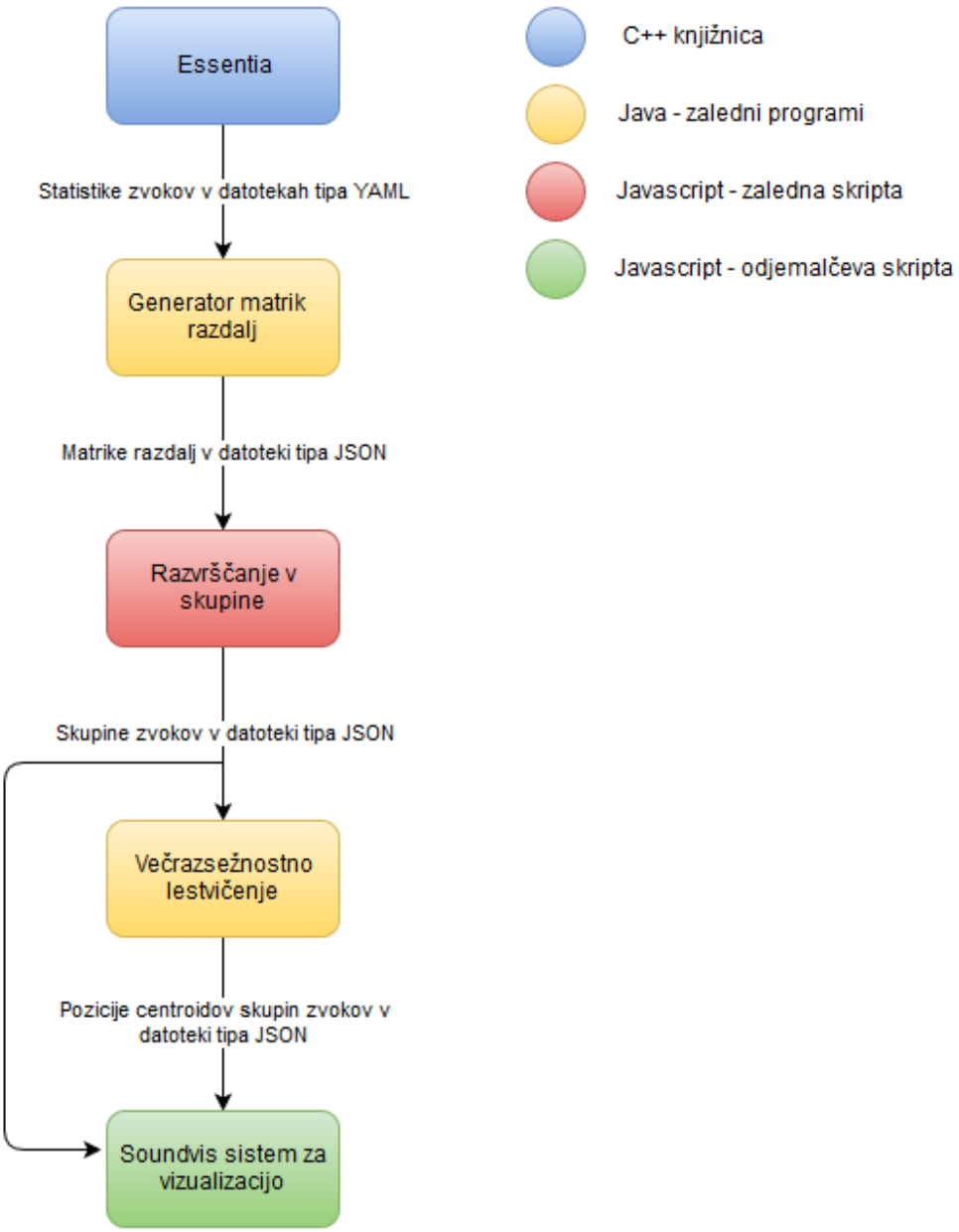
Izvedba sistema za vizualizacijo

3.1 Pregled sistema za vizualizacijo

Preden se spustimo v podrobnosti implementacije sistema za vizualizacijo zbirke zvočnih podatkov, si oglejmo komponente, iz katerih je sistem sestavljen.

Kot je vidno na sliki 3.1 najprej s programsko knjižnico Essentia izračunamo značilke in statistike zvočnih podatkov, ter jih shranimo v datoteke tipa YAML. Te podatke uporabimo v generatorju matrik razdalj, ki izračuna matrike medsebojnih razdalj vseh zvokov. Matrike so velikosti n x n, kjer je n število zvokov. Za vsako skupino značilk izračunamo lastno matriko razdalj. Te matrike se uporabijo v zaledni skripti, ki razvrsti zvoke v skupine po metodi k-means oz. k-najbližjih voditeljev. Skupine zvokov so zapisane v datotekah tipa JSON. Za izračun dvodimenzijskih koordinat uporabimo javansko knjižnico za večdimenzijsko lestvičenje MDSJ. Te koordinate predstavljajo pozicije centroidov skupin zvokov in jih uporabimo kasneje v sistemu za vizualizacijo.

Datoteki s skupinami zvokov ter pozicijami centroidov skupin služita kot vhodni datoteki sistemu za vizualizacijo. Ta sistem, ki teče na odjemalčevi strani, izvede samo vizualizacijo zvočnih podatkov glede na vhodne podatke.



Slika 3.1: diagram komponent sistema za vizualizacijo

3.2 Računanje podobnosti zvokov

V tem podpoglavju je opisan potek izračuna podobnosti med zvoki. Za iskanje podobnih zvokov najprej izračunamo značilke za vse zvoke v zbirki. Na podlagi vrednosti značilk nato zvoke združujemo v skupine podobnih zvokov. Za izračun značilk uporabljamo odprtokodno knjižnico Essentia, ki je dostopna na spletni strani [24], ter za vsak zvok izračunamo določene značilke.

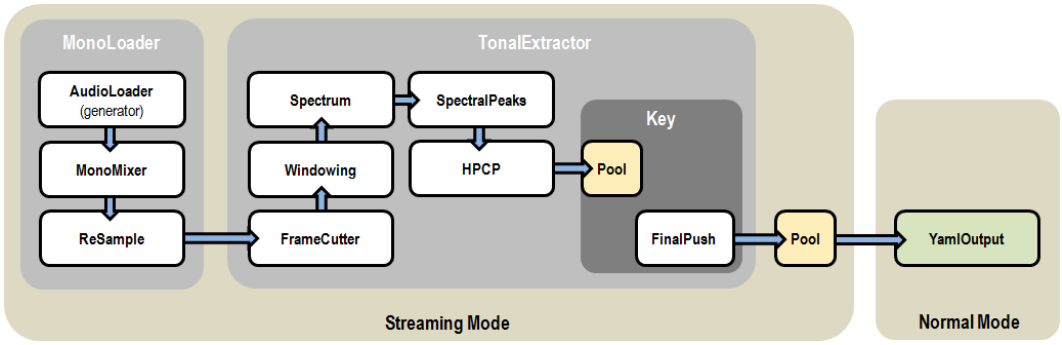
Nekatere značilke opisujejo podatke zgolj s posameznimi vrednosti, pri drugih pa so prisotni tudi statistični deskriptorji, ki omogočijo bolj natančno primerjavo podatkov.

3.3 Pregled knjižnice Essentia

Essentia je odprtokodna knjižnica napisana v jeziku C++, ki jo uporabljamo za zvočno analizo, ter pridobivanje informacij na podlagi zvočnih zapisov [5]. Knjižnica je izdana pod licenco Affero GPLv3 [12] in vsebuje široko zbirko algoritmov za večkratno uporabo, ki so uporabljeni za implementacijo zvočnih vhodno/izhodnih funkcionalnosti. Poleg procesiranja blokov digitalnih signalov in pridobivanja statističnih lastnosti, so ti algoritmi zadolženi tudi za računanje večje zbirke spektralnih, časovnih, tonalnih in visokonivojskih zvočnih deksriptorjev oziroma opisnikov [24].

Kot lahko vidimo na diagramu 3.2, je knjižnica sestavljena iz nekaj povezanih gradnikov. Z uporabo tokovnega načina se uvozijo zvočni podatki, ter se ustrezno vzorčijo. Ti vzorci se uporabijo za računanje značilk, pri tem pa se upoštevajo različne lastnosti vzorcev. Na koncu se rezultati zapišejo v datoteko YAML. YAML je jezik za serializacijo z zapisom, ki je razumljiv tudi človeku [10]. S pomočjo zamikov in drugih ločil se določa hierarhično urejenost med zapisanimi podatki. Jezik YAML je namenjen predvsem shranjevanju podatkov za kasnejšo rabo in s tega vidika je povsem uporaben za zapis rezultatov.

Za Essentio lahko rečemo, da ni programsko ogrodje, pač pa zbirka algorit-



Slika 3.2: predstavitev gradnikov knjižnice Essentia (vir: [25])

mov, ki je zapakirana v knjižnico. Knjižnico Essentia lahko uporabimo na več načinov. Ker ima Pythonovo ovojnico, lahko uporabljamo funkcije knjižnice s pomočjo Python skript. Ena izmed možnosti je tudi uporaba enega izmed že pripravljenih izluščevalcev zvočnih značilk, ki jih lahko zaženemo iz Linuxove ukazne vrstice.

Za naše potrebe vsebuje Bash izluščevalec z nazivom streaming_extractor _freesound [30] vse potrebne značilke za celovito zvočno analizo. Za vsak zvok nam ta izluščevalec vrne datoteko z zbirko značilk zvoka. Nekatere značilke imajo izračunane tudi statistične opisnike, kot so: srednje vrednosti, variance, minimalne in maksimalne vrednosti, srednje vrednosti razlik in podobne. Vse značilke v neki meri pripomorejo k razločevanju zvokov. V naši dvonivojski metodologiji smo za računanje podobnosti uporabili nekaj značilk, ki se med seboj dovolj razlikujejo, da lahko vsaka značilka doprinese k izračunavanju podobnosti. S tem dosežemo, da lahko zvoke celovito primerjamo. Če bi za izračun vzeli same med seboj podobne značilke, bi tako dobili precej pomanjkljive skupine podobnih zvokov, saj bi se lahko zvoki denimo na nekem spektru precej ujemali, ampak hkrati bili po drugih značilnostih precej različni.

V naslednjem poglavju si oglejmo značilke, ki smo jih uporabili za iskanje podobnosti med zvoki. Razdeljene so na več skupin, ki jih lahko v vizualizaciji poljubno otežimo. Oteži imajo vrednosti 0, 0,5 in 1. Matrike razdalj so

izračunane v programu za generiranje matrik razdalj med zvoki (poglavje 4.1) za vsako skupino značilk posebej.

3.4 Pregled skupine značilk za opis disonance

V skupini značilk za opis disonance so značilke, ki jih pridobimo z algoritmom bark bands in značilka, ki opisuje disonanco zvoka. Algoritem bark bands izračuna spektralno energijo v danem številu frekvenčnih pasov [22]. Značilke barkbands_kurtosis, barkbands_skewness in barkbands_spread hranijo vrednosti statistik centralnih momentov spektralnih energij [28]. Značilka barkbands_kurtosis nam pove, kako sploščena je porazdelitve teh centralnih momentov. Značilka barkbands_skewness nam pove kako asimetrična je porazdelitev centralnih momentov spektralnih energij. Značilka barkbands_spread nam pove kako raztegnjena oziroma stisnjena je porazdelitev. S to značilko izvemo ali so porazdelitve bolj ali manj enakomerno porazdeljene po celotnem pasu.

Algoritem za izračun statističnih vrednosti disonance je uporaben za izračun senzorične disonance spektra, ki meri zaznavno grobost zvoka [23].

3.5 Pregled skupine značilk za opis spektra

V skupini značilk za opis sepktra se osredotočamo predvsem na spekter zvoka in na visokofrekvenčno vsebino, ter delež tihih okvirjev v zvočnem posnetku.

HFC je značilka, ki pove količino visokofrekvenčne vsebine v signalu [35] [26]. Značilka spectral_centroid opisuje kje je center teže spektra [28] oziroma za koliko so porazdelitve odmaknjene od spektralnega središča. S to značilko izvemo ali gre za bolj visok ali bolj nizek zvok. Značilka spectral_energy vsebuje statistiko vrednosti energije spektra. Značilka spectral_complexity opisuje kompleksnost spektra glede na število konic v spektru [29]. Te značilke so uporabne za ugotavljanje glasnosti zvokov. Značilka silence_rate_20dB opisuje kolikšen delež zvoka je tišji od 20 dB.

3.6 Pregled skupine značilk za opis zvočne barve

Preglejmo skupino značilk za opis barv zvokov. Naslednje značilke nosijo vektorske vrednosti statističnih opisnikov, zato jih je potrebno med generiranjem matrike podobnosti posebej obravnavati. Algoritem za izračun značilke MFCC izračuna koeficiente MFCC [27], ki se pogosto uporabljajo za klasifikacijo žanra, iskanje informacij v glasbi, razpoznavanje govorca in prepoznavo govora [37]. Koeficienti MFCC opisujejo barvo zvoka. Značilki spectral_contrast in sc_valleys opisujeta značilnosti kontrasta spektra zvoka.

Poglavje 4

Priprava podatkov za vizualizacijo

Preden lahko uporabimo značilke zvokov, njihove statistike in lastnosti, jih moramo obdelati. V nadaljevanju si oglejmo vse preostale komponente zalednega sistema, ki so del naše rešitve za vizualizacijo. Zaledne komponente so zaslužne za računanje oddaljenosti zvokov, kjer so bolj podobni zvoki manj oddaljeni. Zaledne komponente so uporabne tudi za združevanje zvokov v skupine, ter končno pripravo podatkov za prikaz na odjemalčevem spletnem brskalniku. Zaradi časovnih zahtevnosti zalednih algoritmov je smiselno zbrati podatke za nespremenjeno zbirko zvokov le enkrat, ter jih ponovno uporabiti. V primeru da pride do sprememb v zbirki, bodisi se kakšen zvok doda ali odvzame, je potrebno vse sledeče zaledne komponente pognati ponovno, saj moramo pri gručenju upoštevati vse zvoke v prikazani zbirki. Za začetek si oglejmo naslednjo v vrsti zalednih komponent, ki sledi izračunom značilk s knjižnico Essentia. Gre za program za generiranje matrik razdalj.

4.1 Program za generiranje matrik razdalj

Ko so statistike značilk zvokov poračunane, se te vrednosti podajo programu za generiranje razdalj matrik med zvočnimi podatki. Program sem spisal sam v programskem jeziku Java. Pri pisanju programa sem si pomagal s programskima knjižnicama YamlBeans in Gson. Knjižnica YamlBeans je namenjena serializaciji in deserializaciji YAML datotek oziroma njenih podatkov, knjižnica Gson pa je namenjena serializaciji in deserializaciji JSON objektov.

Knjižnica je dostopna na spletni strani [9]. V tem programu je uporabljena za branje podatkov iz YAML datoteke z značilkami zvokov in njihovimi statističnimi vrednostmi. Prebrani podatki se sproti shranjujejo v javanski objekt preslikav (angl. Map). V ta objekt se shranjujejo zaporedne številke zvokov, ki služijo kot identifikacijska števila, ter podatki o nazivih in vrednostih značilk posameznih zvokov.

Knjižnica Gson je dostopna na spletni strani [13]. V tem programu je ta knjižnica uporabljena za shranjevanje izračunanih matrik razdalj med zvoki v datoteko tipa JSON. Podatke shranjujemo v datoteki JSON zaradi združljivosti s skripto za razvrščanje v skupine.

Da lahko zvoke združimo v več skupin med seboj podobnih, moramo najprej vsakega izmed zvokov primerjati z vsemi drugimi. Ko je deserializacija iz datoteke YAML končana, ter so podatki v javanskem objektu preslikav, začnemo z izluščevanjem podatkov tistih značilk, ki jih bomo uporabili pri računanju razdalj med zvoki. To storimo zato, da bomo v nadaljevanju razpolagali z manjšimi objekti. Kasneje od vsake značilke, ki jo bomo uporabili, shranimo naslednje statistične podatke(angleški nazivi so v oklepajih): srednjo vrednost (mean), delta srednjo vrednost (dmean), varianco (var) in delta varianco (dvar). Ker so statistične vrednosti pri nekaterih značilkah shranjene v tabelah, je pri teh značilkah potreben poseben pristop. Pri značilkah, kjer so vrednosti zapisane v decimalnih številih, ta števila izpeljemo v javanski objekt tipa Double, ki hrani vrednost števila s predstavitvijo v plavajoči vejici z dvojno natančnostjo. Pri značilkah, kjer so vrednosti zapisane v tabe-

lah decimalnih števil, te vrednosti ustrezno izpeljemo v tabele objektov tipa Double. Vrednosti teh značilk imajo zaradi drugačne podatkovne strukture tudi v nadaljevanju posebno obravnavo. Ker se ob prebiranju iz datoteke YAML vrednosti statističnih opisnikov obravnavajo kot objekti tipa String, je potrebna izpeljava vrednosti v objekte tipa Double.

Vrednosti statističnih opisnikov so shranjene v objektih tipa Map. Ker za vrednosti ključev uporabljamo ID vrednosti zvokov, ima vsak zvok svojo tabelo vrednosti, kjer so shranjene prej izluščene in izpeljane vrednosti.

Naslednji korak je standardizacija statističnih opisnikov oziroma atributov značilk zvokov. Absolutna vrednost standardizirane vrednosti predstavlja razdaljo med vrednostjo atributa in njegovo pričakovano oziroma srednjo vrednostjo [38]. Sledi izraz za izračun standardizirane vrednosti atributa (tudi z-vrednosti) 4.1.

$$z = \frac{x - \mu}{\sigma} \tag{4.1}$$

Za izračun pričakovane vrednosti in standardnega odklona nekega atributa se upoštevajo vse vrednosti atributa, ki se pojavijo v celotni zbirki zvokov.

Za računanje razdalj med zvoki uporabljamo kosinusno razdaljo. Kosinusne razdalje se izračunajo med vsemi zvoki, s čimer dobimo matrike, kjer vrednost j-tega stolpca i-te vrstice predstavlja kosinusno razdaljo med zvokoma z zaporednimi številkami i in j. Diagonalni elementi, kjer sta vrednosti indeksov i in j enaki, imajo vrednosti 0, saj je zvok sam sebi povsem enak.

Kosinusna razdalja se izračuna na podlagi kosinusne podobnosti, ki nam pove, kakšen je kosinus kota med dvema vektorjema. Ker ima vsak zvok za vsako značilko shranjeno tabelo vrednosti statističnih opisnikov, lahko gledamo na te tabele vrednosti kot na vektorje. Zvoke pa potem primerjamo glede na te vektorje. Kosinusna razdalja ima vrednost med 0 in 1, kjer za razliko od kosinusne podobnosti velja, da višja kot je podobnost med zvokoma, manjša je razdalja oziroma bolj se razdalja približuje vrednosti 0

[34]. V nadaljevanju 4.2 si oglejmo izraz za izračun kosinusne razdalje med dvema vektorjema A in B.

$$razdalja = 1 - podobnost = 1 - \frac{A \cdot B}{\|A\|\|B\|} \tag{4.2}$$

V programu zgeneriramo tri matrike oddaljenosti. Prva matrika je izračunana na podlagi vrednosti značilk za opis disonance. V tej matriki se upoštevajo vrednosti značilk z nazivi: barkband_curtosis, barkband_skewness, barkband_spread in dissonance.

Druga matrika je izračunana na podlagi vrednosti značilk za opis spektra. V tej matriki se upoštevajo vrednosti značilk z naslednjimi nazivi: spectral_centroid, spectral_complexity, spectral_energy, spectral_contrast, hfc in silence_rate_20dB. V tej matriki pridejo do izraza energičnost, lastnosti spektra, količina visokofrekvenčne vsebine, ter delež zvoka ki je tišji od 20dB.

Tretja matrika pa je izračunana na podlagi vrednosti značilk, ki opisujejo zvočno barvo. Ta matrika je izračunana na podlagi značilk mfcc in scvalleys. Ti značilki imata za vrednosti statističnih opisnikov tabele vrednosti, tako da jih obravnavamo s posebej prilagojenimi metodami. Omenjene značilke smo na kratko opisali v poglavju 3.3.

Kosinusne razdalje med zvoki izračunamo za vsako značilko posebej. Za izračun oddaljenosti zvokov v matrikah vzamemo povprečne vrednosti oddaljenosti med vsemi značilkami, ki so zajete v neki matriki. Tako na koncu dobimo 3 matrike velikosti n x n, kjer je n število zvokov v zbirki. Te matrike se shranijo v datoteko tipa JSON, da se lahko uporabijo v naslednji komponenti naše rešitve, kjer se zvoki združujejo glede na vrednosti v matrikah razdalj med zvoki.

4.2 Skripta za razvrščanje zvokov v skupine

V tem poglavju si bomo ogledali, kako zvoke razvrščamo v skupine med seboj podobnih zvokov. Za razvrščanje zvokov v skupine uporabljamo programsko

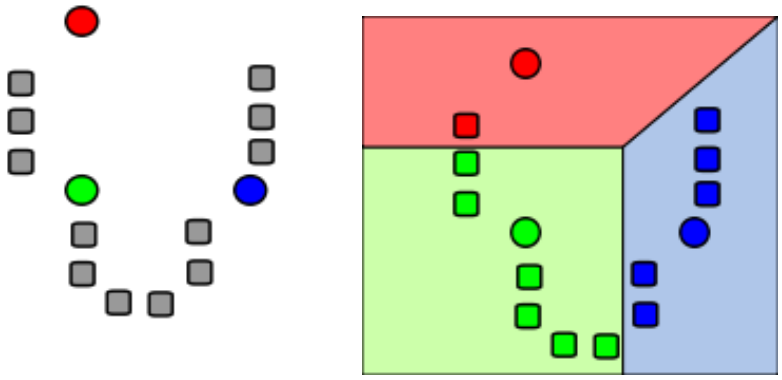
knjižnico Clusterfck, ki je napisana v jeziku JavaScript in je na voljo na [1]. Zvoke razvrščamo po metodi z voditelji (angl. k-means clustering), ki je implementirana v knjižnici Clusterfck. Zvoke razvrstimo v 7 skupin na podlagi sedmih voditeljev, nato pa na teh skupinah ponovno opravimo razvrščanje, da dobimo za vsako skupino še 7 podskupin. S tem dosežemo dvonivojsko hiearhično razvrščenost zbirke zvočnih podatkov.

4.2.1 Razvrščanje z voditelji

Razvršanje z voditelji se pogosto uporablja za analizo gruč in v podatkovnem rudarjenju [36]. Cilj te metode je razvrstiti n elementov v k gruč, kjer vsak element pripada gruči, katere centroid mu je najbližji. Za poenostavitev poglejmo kako poteka standarden algoritem metode za razvrščanje z voditelji na elementih z dvema koordinatama. Na slikah 4.1 lahko vidimo prikaz poteka algoritma. Na teh slikah krogi predstavljajo centroide oziroma voditelje skupin, kvadrati pa predstavnike oziroma elemente skupin. Najprej (slika 4.1a) se določijo začetne pozicije oziroma vrednosti centroidov, ki so lahko določene naključno. Nato (slika 4.1b) se opravi prvo razvrščanje predstavnikov k skupinam, katerih voditelji so predstavnikom najbližji. Sledi (slika 4.1c) ponovno računanje vrednosti centroidov, torej pozicij voditeljev. Na koncu (slika 4.1d) se ponovno opravi razvrščanje v skupine glede na oddaljenosti od voditeljev skupin. Koraka na slikah 4.1b in 4.1c se ponavadi ponavljata, dokler ne pride do konvergence, ko predstavniki ne prehajajo več k drugim skupinam oziroma ko se pozicije voditeljev ne spreminjajo več.

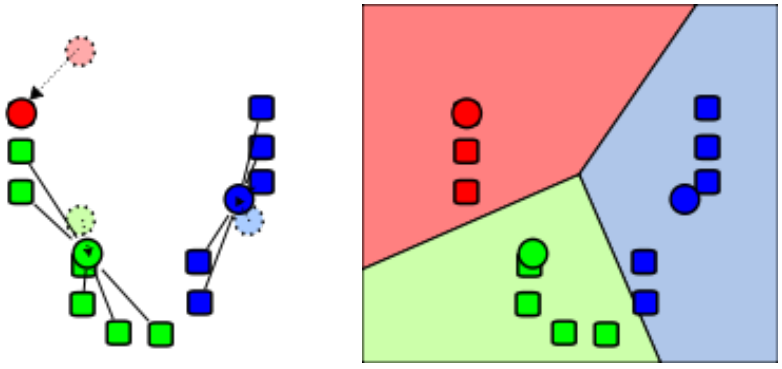
4.3 Računanje centroidov skupin

Za računanje centroidov skupin uporabljamo programsko knjižnico MDSJ, napisano v programskem jeziku Java. Knjižnica je na voljo na spletni strani [33]. Z metodo večrazsežnostnega lestvičenja izračunamo dvodimenzionalni koordinati za vse zvoke. Tako dobimo iz vektorjev velikosti n, vektorje velikosti 2, kjer je n število zvokov. Ti dve koordinati predstavljata osi x in y



(a) začetno stanje

(b) razvrščanje k skupini z najbližjim voditeljem



(c) ponovno računanje voditeljev skupin

(d) ponovno razvrščanje v skupine

Slika 4.1: Demonstracija razvrščanja v skupine po standardnem algoritmu razvrščanja po metodi z voditelji (vir: [36])

v naši vizualizaciji. Kot vhod programa služita dve vrsti podatkov: podatki predstavnikov o pripadnostih skupinam, ter matrike razdalj med zvoki. Skupine in njihove predstavniki so zapisani v lastni datoteki tipa JSON, tako kot matrike razdalj med zvoki. Vrednosti podatkov za obe vrsti so odvisne od uteženosti matrik razdalj. Za hitrejše delovanje v času pregleda vizualizacije se vse matrike razdalj z vsemi možnimi kombinacijami uteži izračunajo vnaprej. Uteži imajo lahko 3 vrednosti: 1, 0,5 ali 0. Pri računanju matrik razdalj je izpuščena tista kombinacija, kjer imajo vse 3 uteži vrednost 0. Podatki za vsako kombinacijo uteži so v lastni datoteki, tako da moramo to upoštevati v programu za računanje centroidov skupin. Za vsako kombinacijo uteži imamo kot vhod v program 2 datoteki tipa JSON. Za branje teh podatkov ponovno uporabimo knjižnico Gson [13]. Za neko kombinacijo uteži se izračuna centroide skupin glede na razvrščenosti zvokov v skupine in oddaljenosti med zvoki. Vektor razdalj vsakega zvoka se z algoritmom večrazsežnostnega lestvičenja pretvori v vektor velikosti 2. Nato se za vsako skupino na 1. nivoju in vse njhove podskupine na 2. nivoju izračuna povprečne vrednosti predstavnikov, katere predstavljajo vrednosti centroidov skupin in podskupin. Ti rezultati se potem shranijo v datoteke tipa JSON, pri tem se centroidi skupin za vsako kombinacijo uteži shranijo v lastno datoteko. Za shranjevanje v datoteke tipa JSON ponovno uporabimo programsko knjižnico Gson.

Poglavje 5

Realizacija vizualizacije

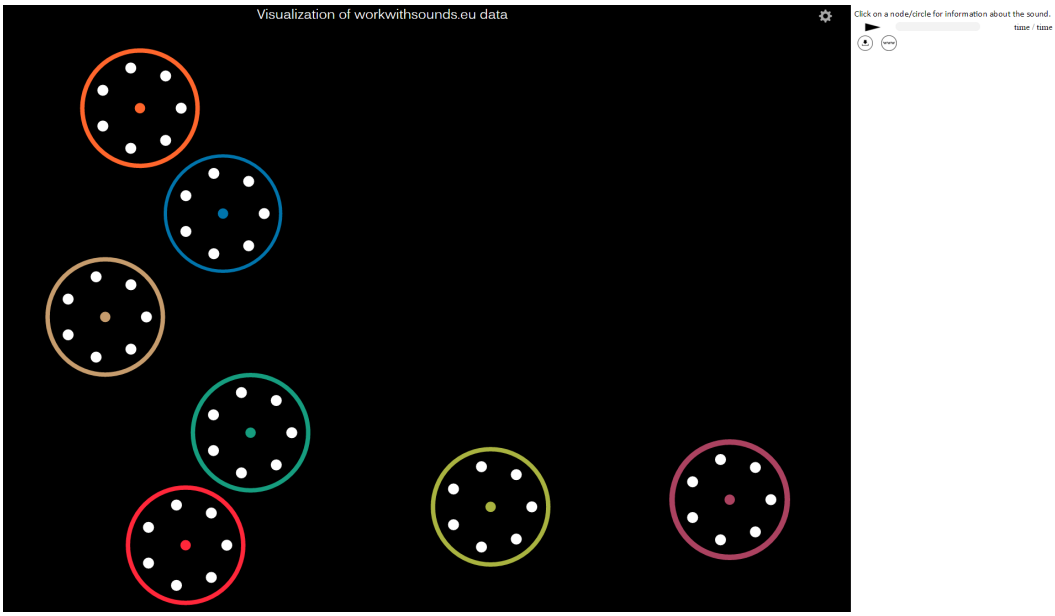
Poglejmo zadnjo in osrednjo komponentno naše rešitve. Gre za sistem za vizualizacijo zvočnih podatkov s pomočjo knjižnice svg.js, ki je namenjena dinamičnemu izrisovanju grafik SVG. Prednost SVG slik oziroma stopnjevanih vektorskih slik je, da se pri povečavi elementa jasnost ohrani, saj je slikovni element zapisan z vektorji.

Najprej preglejmo zaslonske slike in rezultate sistema za vizualizacijo zvočnih podatkov, nato pa sledi še podrobnejši pregled nekaterih komponent sistema.

5.1 Rezultati

Na sliki 5.1 je viden začetni zaslon sistema za vizualizacijo. Na levem delu zaslonske slike so prikazane skupine na prvem nivoju s predstavniki teh skupin. Desni del vizualizacije je namenjen predstavitvi ključnih informacij o zvoku. Med te sodijo ključne besede, lokacija, časovna umestitev naprave in krajši opis zvoka. Zvok časovno umestimo glede na desetletje, v katerem je bila izdelana naprava, ki proizvaja ta zvok. Ker je projekt mednaroden, je vsebina vizualizacije v angleščini, s čimer poskrbimo tudi za možnost uporabe na mednarodnem področju.

Na desnem delu vizualizacije so prisotni tudi: gumb za prenos predvaja-

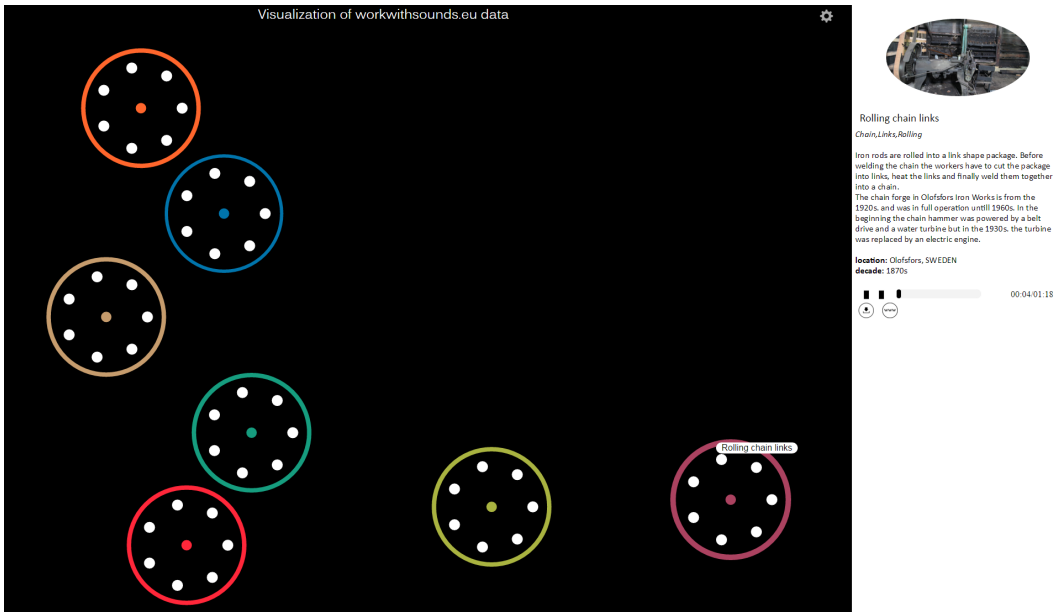


Slika 5.1: zaslonska slika začetnega stanja sistema za vizualizacijo

nega zvoka, predvajalnik zvokov in gumb za dostop do spletne strani zvoka na strani projekta Work With Sounds.

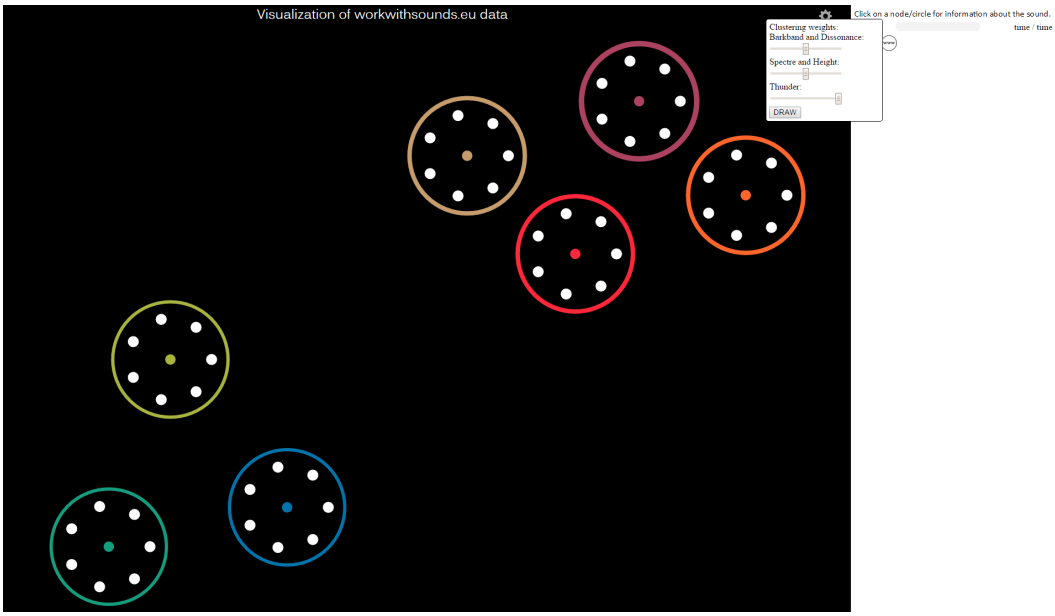
Ob preletu čez točko, ki predstavlja zvok, se nad točko prikaže namig (angl. tooltip) z naslovom zvoka, kar je vidno na sliki 5.2. Namig je prisoten, dokler z miškinim kazalcem ne zapustimo področja točke zvoka. Ob kliku na središče ene izmed skupin, preidemo na drugi nivo vizualizacije, kjer so prikazane podskupine izbrane skupine (slika 5.3). Po teh skupinah lahko preiskujemo tako kot na 1. nivoju, le da so tokrat prisotni vsi zvoki prikazanih skupin, med tem ko so na 1. nivoju prikazani le nekateri predstavniki skupin. Če držimo miško vsaj 1 sekundo na eni od točk, ki predstavljajo zvoke, se bo začel predvajati ustrezen zvok.

V vizualizaciji lahko tudi spremenimo uteži matrik razdalj (slika 5.4). Ob tem dejanju se preberejo podatki iz drugih datotek, nato se vizualizacija znova izriše. Ker so skupine v novih podatkih drugačne, se spremenijo tudi koordinate središč skupin.



Slika 5.2: prelet čez element zvoka za predvajanje in informacije o zvoku

Slika 5.3: podskupine vijolične skupine, kjer predvajamo drug zvok



Slika 5.4: sprememba uteži matrik razdalj in prikaz nove vizualizacije 1. nivoja

5.2 Programska knjižnica svg.js

Programska knjižnica svg.js je dostopna na strani [11] in je izdana pod licenco MIT. S knjižnico je mogoče dinamično in pregledno upravljati s SVG elementi, ter poskrbeti za odzive na upravljanje uporabnika. Med pogostejšimi uporabnikovimi akcijami za upravljanje so denimo klik na element, dvojni klik na element, prehod s kazalcem nad komponento in podobni. Vsak SVG element ima v tej knjižnici svoj ID, tako da se lahko posebej upravlja za vsakim elementom. Možno pa je upravljati tudi s skupino elementov, ali pa denimo odstraniti vse elemente, ki so trenutno prisotni.

5.3 Posredovanje podatkov sistemu za vizualizacijo

Strežnik poleg same skripte napisane v jeziku JavaScript, pošlje še vse ustrezne podatke o zvokih in gručah glede na izbrane uteži. Poleg datotek uporabljenih knjižnic, se posredujejo informacije o zvokih, skupine s predstavniki ter koordinate centroidov skupin.

Datoteke s podatki se prenesejo z zahtevkom Ajax s pomočjo knjižnice jQuery, ki je dostopna na [17]. Knjižnica jQuery je napisana v jeziku JavaScript in poenostavi pisanje skript za odjemalce. Ko so vse datoteke prenešene, ter so na voljo vsi potrebni podatki, se vizualizacija prikaže.

5.4 Dvonivojskost

Sedaj si podrobneje oglejmo dvonivojskost vizualizacije. Zaradi večjega števila zvokov smo izvedli vizualizacijo na dveh nivojih. Na prvem oziroma začetnem nivoju so prikazane skupine z nekaj predstavniki njihovih podskupin. Širina obrobe skupin je odvisna od števila vseh predstavnikov v podskupinah. Več jih je, širša je obroba. S tem dobimo občutek o velikosti skupin.

Ko izberemo in preidemo v eno izmed skupin na prvem nivoju, se prikažejo vse njene podskupine na drugem nivoju. Pri tem je premer obrobe skupine odvisen od števila predstavnikov v skupini. Na drugem nivoju so prikazani vsi predstavniki podskupin. Prehod na prejšnji nivo je možen s klikom na element domov, torej element s hiško, ali pa z dvojnim klikom na barvno ozadje.

Z nekaj prilagajanji bi lahko naredili vizualizacijo tudi večnivojsko, vendar bi se na visokih nivojih lahko že precej razbile skupine, ki bi nam sicer bile zanimive v celoti. Prav tako bi postala vizualizacija nekoliko manj pregledna, ter bi zahtevala od uporabnika več akcij za analizo podatkov. Večnivojska verzija sistema za vizualizacijo bi tako bila primerna le pri zelo velikem številu zvočnih podatkov, ko bi bilo v zbirki več tisoč zvokov.

5.5 Barve in generiranje barv podskupin

Za razločevanje med skupinami ima vsaka skupina svojo barvo za obarvanje SVG komponent. S temi barvami se obarvajo centri ter obrobe skupin, na drugem nivoju pa še barvno ozadje. Uporabili smo barve, ki so druga od druge čim bolj oddaljene na barvnem spektru, tako da jih lažje razločimo.

Barve prikazane na prvem nivoju so izbrane vnaprej, ter so shranjene v hash zapisu v skripti sistema za vizualizacijo. Pri tem ima vsak izmed kanalov RGB določeno 32-bitno vrednost.

Za prikaz barv na drugem nivoju pretvorimo zapis barv v RGB prostoru v zapis barv v barvnem prostoru HSL. Uporabna lastnost tega barvnega prostora je kanal L (Lightness oziroma svetlost), s katerim lahko osvetlimo ali potemnimo barvo. To uporabimo za prikaz barv na drugem nivoju, tako da ima vsaka podskupina svoj odtenek barve, s čimer jih lažje ločimo drugo od druge. Ker knjižnica svg.js še ne podpira zapisa barv v barvnem prostoru HSL, sproti pretvarjamo barve med prostoroma RGB in HSL. Knjižnici za izris SVG komponent tako vsakič podamo zapis barv v prostoru RGB.

5.6 Podatkovne strukture

Informacije in podrobnosti zvokov so zapisane v objektu JSON. Večina podatkov za podporo izvedbi vizualizacije je shranjenih v dvodimenzionalnih tabelah. SVG elementi vizualizacije so shranjeni v knjižnici za vizualizacijo svg.js, njihove reference pa so shranjene v tabelah naše skripte. V naši vizualizaciji imamo več različnih skupin SVG elementov, med katerimi so: obrobe skupin, središča skupin, ozadja, točke zvokov in gumb za prehod na 1. nivo. V sistemu za vizualizacijo je za hrambo referenc elementov vsake od teh skupin na voljo lastna dvodimenzionalna tabela. Na ta način se lahko upravlja z vsako skupino elementov posebej, pri tem pa ni potrebno iskanje po celotnem seznamu elementov.

Za bolj pretočen prehod na prejšnji nivo ohranjamo komponente prejšnjega nivoja v pomnilniku. Tako se denimo ozadje postopoma pomanjša in prelije

v središče skupine, ki je bila izbrana na prejšnjem nivoju.

Vhodne koordinate centroidov skupin se preslikajo na območje vrednosti med 0 in 1, kjer je 1 najvišja vrednost koordinate. Ob izrisu se te vrednosti pretvorijo v vrednosti med 0 in številom pikslov na x ter številom pikslov na y osi. Ti števili sta odvisni od velikosti HTML okvirja, v katerem se izrisujejo SVG komponente.

5.7 Premik skupin za reševanje prekrivanja

Za rešitev prekrivanja SVG komponent, jih drugo od druge pred prikazom ustrezno odmaknemo. Pri tem moramo paziti, da katera od komponent ne zaide izven vidnega območja. Preden izvedemo premik skupine, preverimo če bi ob premiku prišlo do trka z mejami vizualizacije. Sledi psevdokoda algoritma za zaznavanje prekrivanj.

$xVal \leftarrow clustCentroidX + xDist + radius$
$yVal \leftarrow clustCentroidY + yDist + radius$
if $xVal \geq 0.98$ OR $xVal \leq 0.02$ **then**
    $collisionX \leftarrow true$
end if
if $yVal \geq 0.98$ OR $yVal \leq 0.02$ **then**
    $collisionY \leftarrow true$
end if

Če je trk zaznan na kateri od osi, potem na tisti osi ne izvedemo premika. V primeru da sta zaznana trka na obeh oseh, premika skupine ne izvedemo.

Premik skupine skušamo izvesti, kadar zaznamo trk med dvema skupinama na nivoju. Pri tem upoštevamo koordinate centroidov skupin, ter polmere njihovih obrob. Zaznavi trkov je namenjena funkcija detectOverlaps, ki glede na vhodne podatke izračuna razdaljo med skupinami, ter vrne vrednost true, če je zaznan kakšen trk. Vhodna podatka sta dve tabeli, in sicer: enodimenzionalna tabela polmerov obrob skupin, ter dvodimenzionalna tabela koordinatnih vrednosti centroidov skupin.

Premik izvedemo nad tistimi skupinami, ki se prekrivajo. Premikamo po dve skupini naenkrat. Najprej izračunamo razdaljo, za katero se skupini prekrivata. To izračunamo tako, da od seštevka polmerov obrob obeh skupin odštejemo razdaljo med obema središčema. Nato preverimo v kakšnem razmerju sta obe središči glede na koordinati. Če sta koordinati poravnani po kateri izmed osi, potem ju pomaknemo samo po preostali osi in sicer za polovico razdalje prekrivanja. V primeru da sta oba para koordinat enaka, potem eno izmed skupin premaknemo stran od druge. Kadar skupini nimata enakih vrednosti na nobeni izmed koordinat, potem ju premaknemo po obeh koordinatah. V tem primeru vsako od središč premaknemo v nasprotno smer za polovico razdalje prekrivanja skupin.

To premikanje ponavljamo dokler zaznamo prekrivanje. Tako se v nekaj iteracijah razreši prekrivanje med skupinami.

5.8 Izris skupine

Točke, ki predstavljajo zvoke, so izrisane okrog središča skupine, ki ji pripadajo. Kot je vidno na sliki 5.1, so točke zvokov izrisane na navideznem krogu, ki obdaja središče. Torej so vse točke v skupini enako oddaljene od središča. Koordinate posameznih točk se izračunajo po naslednjem algoritmu, kjer so vhodni podatki: koordinati središča skupine, število zvokov v skupini, ter polmer navideznega kroga.

$stepRad \leftarrow 2 * Math.Pi / numOfMembers$
for i in $[0, numOfMembers - 1]$ **do**
    $x \leftarrow centerX + radius * Math.cos(i * stepRad)$
    $y \leftarrow centerY + radius * Math.sin(i * stepRad)$
    $positions[i] \leftarrow x, y$
end for
return *positions*

S tem so točke v skupini postavljene na navidezni krog, obenem pa so med seboj enako oddaljene. Za večje število točk je potrebno povečati polmer

navideznega kroga, da ne pride do prekrivanja.

Ko imamo izračunane koordinate vseh točk v skupini, lahko izrišemo vse SVG elemente skupine. Poleg točk sta med elementi vsake skupine še središče in obroba skupine.

Poglavje 6

Sklepne ugotovitve

Vizualizacija podatkov je pomembna za uporabnikovo interpretacijo razvrstitve in združevanja podatkov. S tem diplomskim delom smo preučili vizualizacijo podatkov, ter skušali kar najbolje implementirati lasten večnivojski sistem za vizualizacijo zbirk zvočnih podatkov, ki je pregleden in enostaven za uporabo. Naš sistem za vizualizacijo je dvonivojski, zaradi česar lahko prikažemo večje zbirke zvočnih posnetkov. Na enonivojskih vizualizacijah je veliko težje pregledno prikazati podatke večje zvočne zbirke, v kateri je vsaj nekaj sto posnetkov.

Zvočni posnetki vizualizacije se gručijo na dveh nivojih. Gručenje zvočnih posnetkov poteka na podlagi podobnosti med zvoki. Za izračun podobnosti uporabljamo zvočne značilke, izračunane s knjižnico za analizo zvokov Essentia. Značilke zvokov so združene v 3 skupine . Vrednosti značilk uporabimo za generiranje matrik razdalj med zvoki, kjer nižje vrednosti v matrikah pomenijo večjo podobnost med zvokoma. Vsaki skupini značilk izračnamo lastno matriko razdalj med vsemi zvoki.

Na 1. nivoju vizualizacije so prikazane skupine zvokov, ter predstavniki njihovih podskupin. Ko izberemo eno izmed skupin 1. nivoja, preidemo na 2. nivo vizualizacije, kjer se prikažejo vse podskupine izbrane skupine. Na 2. nivoju so prikazani vsi zvočni posnetki teh podskupin. Vizualizacijo lahko prilagajamo s spremembami uteži matrik razdalj med zvoki. Tako

lahko damo pri analizi večji pomen eni ali dvema skupinama značilk. Prav tako lahko katero izmed skupin značilk povsem ignoriramo. Ob spremembi uteži se zaradi sprememb v skupinah izvede ponoven izris vizualizacije. Na obeh nivojih vizualizacije je možen pregled informacij o zvokih in predvajanje izbranih zvočnih posnetkih.

Naš sistem za vizualizacijo ima še precej možnosti za nadgradnje in razširitev uporabnosti. V prihodnosti bi lahko projekt razširili v smeri bolj raznolike uporabe in podrobnejšega upravljanja z vizualizacijo. Vizualizacijo bi lahko razširili s celovito spletno storitvijo za analizo poljubnih zvočnih zbirk. Tako bi lahko uporabniki preiskovali lastne zvočne zbirke, ki niso javno dostopne. Ker bi za izračun podobnosti zvokov v zalednem sistemu bilo potrebnega nekaj časa, bi lahko izvedli sistem za obveščanje uporabnikov o stanju obdelave zvočne zbirke. Ta sistem bi obvestil uporabnika, ko bi bila zvočna zbirka pripravljena za pregled. Za ta namen bi bilo dobro zaledne komponente, ki napisane v jeziku JavaScript, prenesti v javansko okolje. Tako bi lažje izvedli komunikacijo med zalednimi komponentami. Ko bi uporabnik pregledal zbirko, bi jo lahko izbrisal iz strežnika, ali pa jo pustil za kasnejšo analizo. V tem primeru bi se vsi podatki, ki so potrebni za izvedbo vizualizacije, ohranili na strežniku. Možne so tudi manjše razširitve našega sistema za vizualizacijo, denimo razširitev z dodatnimi skupinami značilk. Lahko bi tudi nadgradili našo dvonivojsko vizualizacijo v večnivojsko vizualizacijo.

Literatura

[1] Heather Arthur. Clusterfck. `https://github.com/harthur/clusterfck`. [Online] [Dostopano 6.3.2016].

[2] V. Batagelj, F.J. Brandenburg, W. Didimo, G. Liotta, P. Palladino, and M. Patrignani. Visual analysis of large graphs using (x,y)-clustering and hybrid visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 17(11):1587–1598, Nov 2011.

[3] R.A. Becker, S.G. Eick, and A.R. Wilks. Visualizing network data. *Visualization and Computer Graphics, IEEE Transactions on*, 1(1):16–28, Mar 1995.

[4] G. Bisson and R. Blanch. Improving visualization of large hierarchical clustering. In *Information Visualisation (IV), 2012 16th International Conference on*, pages 220–228, July 2012.

[5] D. Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, P. Herrera, O. Mayor, Gerard Roma, J. Salamon, J. Zapata, and Xavier Serra. Essentia: an audio analysis library for music information retrieval. In *International Society for Music Information Retrieval Conference (ISMIR'13)*, pages 493–498, Curitiba, Brazil, 04/11/2013 2013.

[6] M. Chen, D. Ebert, H. Hagen, R.S. Laramee, R. van Liere, K.-L. Ma, W. Ribarsky, G. Scheuermann, and D. Silver. Data, information, and knowledge in visualization. *Computer Graphics and Applications, IEEE*, 29(1):12–19, Jan 2009.

[7] W. de Leeuw, P.J. Verschure, and R. van Liere. Visualization and analysis of large data collections: a case study applied to confocal microscopy data. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):1251–1258, Sept 2006.

[8] E. Di Giacomo, W. Didimo, L. Grilli, and G. Liotta. Graph visualization techniques for web clustering engines. *Visualization and Computer Graphics, IEEE Transactions on*, 13(2):294–304, March 2007.

[9] EsotericSoftware. YamlBeans GitHub. `https://github.com/EsotericSoftware/yamlbeans`. [Online] [Dostopano 4.3.2016].

[10] C. Clarke Evans. YAML: YAML Ain't Markup Language. `http://yaml.org/`, 2001. [Online] [Dostopano 1.3.2016].

[11] Wout Fierens. Svg.js. `http://svgjs.com/`. [Online] [Dostopano 9.3.2016].

[12] Inc. Free Software Foundation. GNU Affero General Public License. `http://www.gnu.org/licenses/agpl.html`, 2007. [Online] [Dostopano 29.2.2016].

[13] Google. google-gson. `https://github.com/google/gson`. [Online] [Dostopano 4.3.2016].

[14] Thomas Grill and Arthur Flexer. Visualization of perceptual qualities in textural sounds. pages 1–8.

[15] P. Held and R. Kruse. Analysis and visualization of dynamic clusterings. In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pages 1385–1393, Jan 2013.

[16] hint.fm. Wind map. `http://hint.fm/wind/`, 2008. [Online] [Dostopano 29.2.2016].

[17] The jQuery Foundation. jQuery. `http://jquery.com/`. [Online] [Dostopano 9.3.2016].

[18] D.A. Keim. Information visualization and visual data mining. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):1–8, Jan 2002.

[19] P. Knees, M. Schedl, T. Pohle, and G. Widmer. Exploring music collections in virtual landscapes. *MultiMedia, IEEE*, 14(3):46–54, July 2007.

[20] Anita Lillie. MusicBox: Mapping and visualizing music collections. `http://thesis.flyingpudding.com/`. [Online] [Dostopano 14.3.2016].

[21] Fabian Moerchen, Alfred Ultsch, Mario Noecker, and Christian Stamm. Databionic visualization of music collections according to perceptual distance. pages 1–8.

[22] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. BarkBands. `http://essentia.upf.edu/documentation/reference/streaming_BarkBands.html`. [Online] [Dostopano 3.3.2016].

[23] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. Dissonance. `http://essentia.upf.edu/documentation/reference/streaming_Dissonance.html`. [Online] [Dostopano 3.3.2016].

[24] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. ESSENTIA. `http://essentia.upf.edu/`. [Online] [Dostopano 29.2.2016].

[25] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. Essentia (2).png. `http://essentia.upf.edu/sites/default/files/essentia%20%282%29.png`. [Online] [Dostopano 29.2.2016].

[26] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. HFC. `http://essentia.upf.edu/documentation/reference/streaming_HFC.html`. [Online] [Dostopano 3.3.2016].

[27] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. MFCC. http://essentia.upf.edu/documentation/reference/streaming_MFCC.html. [Online] [Dostopano 3.3.2016].

[28] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. Music extractor. http://essentia.upf.edu/documentation/streaming_extractor_music.html. [Online] [Dostopano 3.3.2016].

[29] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. SpectralComplexity. http://essentia.upf.edu/documentation/reference/streaming_SpectralComplexity.html. [Online] [Dostopano 3.3.2016].

[30] Universitat Pompeu Fabra Barcelona MTG Music Technology Group. Using executable extractors out-of-box. http://essentia.upf.edu/documentation/extractors_out_of_box.html. [Online] [Dostopano 2.3.2016].

[31] CACM Staff. Visualizations make big data meaningful. *Commun. ACM*, 57(6):19–21, June 2014.

[32] Stephanie and Tony at R2D3. A Visual Introduction to Machine Learning. http://www.r2d3.us/visual-intro-to-machine-learning-part-1/. [Online] [Dostopano 29.2.2016].

[33] Christian Pich University of Konstanz. MDSJ – Multidimensional Scaling for Java. http://algo.uni-konstanz.de/software/mdsj/. [Online] [Dostopano 6.3.2016].

[34] Wikipedia users. Cosine similarity. https://en.wikipedia.org/wiki/Cosine_similarity. [Online] [Dostopano 6.3.2016].

[35] Wikipedia users. High frequency content measure. https://en.wikipedia.org/wiki/High_frequency_content_measure. [Online] [Dostopano 3.3.2016].

[36] Wikipedia users. k-means clustering. `https://en.wikipedia.org/wiki/K-means_clustering`. [Online] [Dostopano 8.3.2016].

[37] Wikipedia users. Mel-frequency cepstrum. `https://en.wikipedia.org/wiki/High_frequency_content_measure`. [Online] [Dostopano 3.3.2016].

[38] Wikipedia users. Standard score. `https://en.wikipedia.org/wiki/Standard_score`. [Online] [Dostopano 5.3.2016].

[39] Jiajun Zhu and Lie Lu. Perceptual visualization of a music collection. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 1058–1061, July 2005.

[40] Liang Zhu, Bing fang Wu, Yue min Zhou, Xin hui Ma, and Lei dong Yang. Research on eco-environmental data visualization for three gorges project. In *Information Engineering and Electronic Commerce, 2009. IEEC '09. International Symposium on*, pages 590–593, May 2009.